

# Archlinux 下 ICC+MKL 方案编译安装 GAMESS

作者: xiooli

邮箱: <xiooli[at]yahoo.com.cn>

网站: <http://joolix.com>

此文系根据自己实际操作写就, 其中参考了 Sobereva 和王涛的文章(见参考文献)在此表示感谢。

本文依据 CC 协议发布, 在保留原作品的署名的情况下你可以自由转载或修改再发布(仅限非商业用途)

---

## 正文

### 一, 机器配置

型号: lenove ThinkCenter M8000t

主板: Intel Q45

CPU: Intel(R) Core(TM)2 Quad CPU Q9550 @ 2.83GHz (四核)

内存: DDR3 1066MHz 2G X 2

硬盘: SATA 500G 7200RPM

### 二, 系统及编译环境

系统: Archlinux x86\_64

内核: 2.6.32-ARCH

SHELL:

bash: version 4.0.35(2)-release

这是用户的默认登录 shell, 编译操作基本上都是在 bash 里面完成的

csh: tcsh 6.17.00 (Astron) 2009-07-10

因 gamess 的很多脚本是用 csh 写成, 故而需要安装 csh (包名是 tcsh)

C 编译器:

GCC: 4.4.2 20091208 (prerelease)

GCC 是默认的编译器, 可以从你发行版的软件源里面安装

ICC: Version 11.1

intel 的 CPU 用 ICC 可以优化执行性能, 其他 CPU 的同学可以不必使用 ICC 需要去 intel 的官网下载安装, 个人和非商业使用只需注册一下, 免费使用

---

### 设置开始

ICC 一般是安装到 /opt 里面去的, 系统并不能直接找到其可执行文件所以需要将其 bin 目录加到 PATH 环境变量里面去, 以我为例:

在 /opt/intel/Compiler/11.1/056/bin/ 下面有两个文件一个目录(依系统位数的不同, 目录的名字可能不一样): intel64/, iccvars.csh 和 iccvars.sh, 你需要在 /etc/profile.d/ 下面建立一个脚本, 名字随意, 比如 iccvars.sh 其内容如下:

```
#!/bin/bash
```

```
source /opt/intel/Compiler/11.1/056/bin/iccvars.sh intel64
```

最后的那个参数 intel64 依你的系统不同而不同, 应该和你在

/opt/intel/Compiler/11.1/056/bin/ 中看的的目录名字相同, 然后赋可执行权限:

```
sudo chmod a+x /etc/profile.d/iccvars.sh
```

注意, 这样只会在重启后生效, 若要当时生效, 须执行一下:

```
./etc/profile.d/iccvars.sh 或 source /etc/profile.d/iccvars.sh
```

此外, 为了让程序(这里是 icc 等)找到它们所依赖的库, 须得将 icc 的 lib 路径加入到 /etc/ld.so.conf

以我为例：

在 `/etc/ld.so.conf` 里面加上一行：

```
/opt/intel/Compiler/11.1/056/lib/
```

然后以 root 权限运行 `ldconfig`

ICC 的设置到此结束

-----设置结束-----

**FORTRAN 编译器：**

IFORT: Version 11.0

这个也是 intel 的编译器，同样需要去其官方注册下载，个人版也是免费的 当然你也可以使用 g95 等开源的 FORTRAN 编译器

-----设置开始-----

IFORT 同样也是安装到 `/opt` 里面去的，所以也需要将其 `bin` 目录加到 `PATH` 环境变量里面去

以我为例：

在 `/opt/intel/Compiler/11.0/074/bin/` 下面有两个文件一个目录（依系统位数的不同，目录的名字可能不一样）：`intel64/`，`ifortvars.csh` 和 `ifortvars.sh`

你亦需要在 `/etc/profile.d/` 下面建立一个脚本，比如 `ifortvars.sh` 其内容如下：

```
#!/bin/bash
```

```
source /opt/intel/Compiler/11.0/074/bin/ifortvars.sh intel64
```

最后的那个参数 `intel64` 依你的系统不同而不同，应该和你在

`/opt/intel/Compiler/11.0/074/bin/` 中看的的目录名字相同，然后赋可执行权限：

```
sudo chmod a+x /etc/profile.d/ifortvars.sh
```

注意，这样只会在重启后生效，若要当时生效，须执行一下：

```
./etc/profile.d/ifortvars.sh 或 source /etc/profile.d/ifortvars.sh
```

此外，为了让程序（这里是 `icc` 等）找到它们所依赖的库，须得将 `ifort` 的 `lib` 路径加入到 `/etc/ld.so.conf` 里面。

以我为例：

在 `/etc/ld.so.conf` 里面加上一行：

```
/opt/intel/Compiler/11.0/074/lib/
```

然后以 root 权限运行 `ldconfig`

IFORT 的设置到此结束

-----设置结束-----

**数学库：**

MKL: Version 10.2.3.029

这个是 intel 的数学核心函数库，据说使用 `ifort` 编译器结合 `MKL` 数学库可以使代码执行速度大大提升。当然你如果不使用 `MKL` 的话会默认使用 `BLAS` 这个库。AMD CPU 的同学可以考虑使用 `ACML` 这个数学库。同样，`MKL` 也可以到 intel 的官网上面申请免费版本

-----设置开始-----

`MKL` 的设置相对简单，因为其只是一些库，没有可执行文件，所以只需要将 `MKL` 库的位置加到 `ld.so.conf` 里面即可。在 `/etc/ld.so.conf` 里面加上一行：

```
/opt/intel/mkl/10.2.3.029/lib/em64t/
```

然后以 root 权限运行 `ldconfig`  
IFORT 的设置到此结束

-----设置结束-----

### 三, 具体操作过程

在配置好了上面的编译环境以后便可以进行后续的编译过程了。

首先, 当然是准备 gamess 的源码了, 去 gamess 的网站上面填写申请, 一般一到两天后会批下来, 然后会邮件通知你下载地址, 用户名和密码, 自己去拖下来, 压缩包的大小约 10M 左右, 我下载的版本是: GAMESS version January 12, 2009 R3 for 64 bit (x86\_64 compatible) under Linux with gnu compilers

然后准备一个安装目录 (我将其安放在 `/home/gamess` 里面), 将压缩包解压到安装目录下面。好了, 可以看到几个可执行文件:

`compall`, `comp`, `lked`, `runall` 和 `rungms`

-----修改脚本-----

我们接下来将编辑这几个文件以使其符合我们自己的实际情况

以我为例:

`compall`:

```
line 16: set TARGET=linux-ia64
line 17: set /home/gamess
line 70: if ($TARGET == linux-ia64) set CCOMP='icc' # 没有 icc 的可以继续 gcc
```

`comp`:

```
line 15: set TARGET=linux-ia64
line 16: set /home/gamess
line 109:if ($TARGET == linux-ia64) set BLAS3=/opt/intel/mkl/10.2.3.029/lib/em64t/
# 如果你没有安装 MKL 的话就不要动上面这行
# 由于我选择的 TARGET=linux-ia64, 在 line 1507 可以看到已经是默认用了 ifort 作为 FORTRAN
# 的编译器, 如果你的 TARGET 是其他的而又想用 ifort 的话, 可以去对应的地方 (即
# if ($TARGET == xxx) then 下面将 FORTRAN 编译器改成 ifort)
```

`lked`:

```
line 18: set TARGET=linux-ia64
line 19: chdir /home/gamess
line 498:if ($TARGET == linux-ia64) then # 若使用 MKL 则在下面添加一行
line 499: setenv MKLPATH /opt/intel/mkl/10.2.3.029/lib/em64t
# 看它的脚本, 理论上来说不添加这行也是可以找到 MKL 的, 不过在我这里报错说
# 找不到 MKL 里面的某个 so, 所以还是加上比较保险
```

`rungms`:

```
line 60: set SCR=/home/gamess/scr/$USER
# 存放运算临时文件的地方, 自己喜欢放哪里放哪里, 如果你内存够大的话也可以放在
# /dev/shm (实际是个在内存里面的目录)
将所有的 ~mike/gamess 换成你的 gamess 安装目录 (我的是 /home/gamess) $USER/scr 替换成 $SCR
你可以通过运行下面命令来一次性完成此工作:
```

```
sed -i 's|~mike/gamess|/home/gamess|g;s|~$USER/scr|$SCR|g' ./rungms
```

如果你想使用 SMP 方式并行多核运算的话如果是双核机器则修改这里:

```
line 497: case br.msg.chem.iastate.edu: # 下面添加一行
line 498: case YOUR-HOST-NAME: # YOUR-HOST-NAME 是运行 hostname 出来的结果
line 499:if ($NCPUS > 2) set NCPUS=2
```

如果是四核机器则修改这里:

```
line 490: case sc.msg.chem.iastate.edu: # 下面添加一行
line 491: case YOUR-HOST-NAME: # YOUR-HOST-NAME 是运行 hostname 出来的结果
line 492:     if ($NCPUS > 4) set NCPUS=4
runall:
line 13: chdir /home/games
```

-----编译 actvte-----

下面我们要编译链接激活 games 代码的程序 actvte (下面命令均在终端下输入):

```
cd /home/games/tools/
# 视你 games 源文件存放地点的不同而 cd 到不同的目录
cp actvte.code actvte.f
sed -i 's|^\*UNIX|    |g' actvte.f
# 把以 *UNIX 的开头用四个空格替换掉, 即取消注释, 亮出 UNIX-LIKE 系统的激活代码
ifort -o actvte.x -Vaxlib actvte.f
# 或 g95 -o actvte.x actvte.f 如果你没有 ifort 的话。成功生成了 actvte.x 以后可以将 actvte.f 删除
```

-----开始编译-----

然后便可以开始编译了, 在终端运行:

```
cd /home/games
./compall
# 如果你想看着它编译的话, 或者让其在后台运行
./compall &> compall.log &
# 上面两条命令选一即可, 无须都执行
# 经过一段取决于你机器性能的时间后 (我这里大约用了 20 分钟) 主要编译便完成了
```

其后编译 games 的分布式数据接口 (DDI) 消息传递库, 无论是否打算并行计算都必须编译, 否则下面 lked 步骤通不过。假设现在主机名是 YOUR-HOST-NAME (用 hostname 命令可以看到当前主机名), 就在 /etc/hosts 里面填上一行当前主机的IP地址和主机名

例如: 192.168.153.3 YOUR-HOST-NAME # [参考文献 1]

命令:

```
sudo su
echo "192.168.153.3 YOUR-HOST-NAME" >> /etc/hosts
```

进入 /home/games/ddi 目录, 修改 compddi

```
compddi:
line 18: set TARGET = linux-ia64
line 62: set MAXCPUS = 4
line 63: set MAXNODES = 2
# set MAXCPUS 和 set MAXNODES 后面设成你的实际情况, 前者代表每个节点中最多包含几个核心 (每个节点内可以
# 以SMP方式并行的核心数目), 后者代表最多有几个节点, 它们设的都可以比实际情况多 [参考文献 1]
```

修改 /home/games/ddi/src/std\_system.c

```
std_system.c
line 26:     struct hostent *hp; # 在其后插入一行
line 27:     name = "YOUR-HOST-NAME";
# 之所以要加入这么一行是因为 GAMESS 的 bug, 会将主机名强行认作为 localhost 而不是实际主机名, 导致运行时
# 提示 TCP error 之类错误 [参考文献 1]
```

最后运行命令:

```
./compddi
mv ddickick.x ..
```

最后编译图形应用程序

```
cd /home/gamess/graphics
```

编辑 complink 文件

complink:

```
line 15: chdir /home/gamess/graphics
line 18: set TARGET=linux-pc
line 31: if ($TARGET == linux-pc) set FORT='ifort -O2'
```

之后运行 ./complink 来编译

-----编译结束-----

链接可执行文件

命令：

```
cd /home/gamess
./lkd gamess &> lkd.log
```

如果不出问题的话最终会在当前目录下生成一个 gamess.00.x 这样的可执行文件，至此我们的编辑工作就全部完成了，下面就是测试环节，在 /home/gamess 目录下运行 ./runall 将会对编译好的 GAMESS 程序进行测试，自动运行 44 个测试文件，将在当前目录下得到一批 exam??.log 文件。修改 /home/gamess/tools/checktst 目录下的 checktst 文件：

checktst:

```
line 9: set GMSPATH=/home/gamess
```

然后运行此文件，出现提示时输入 /home/gamess，会将所得 log 文件的结果与标准结果相对比检查任务是否已正常结束，以及计算误差是否超过阈值。对于 failed 的任务，检查相应的 log 文件除掉毛病后，把 scr 目录下对应文件删掉，若要单独运行一个测试比如 exam42 可以使用命令：

```
cd /home/gamess
./rungms exam42 00 4 &>exam42.log
# rungms 的参数意义为：JOB=$1 (输入文件 xxx.inp, 可以不输入扩展名); VERN0=$2 (lkd 一步的 revision 数字, 默认 00); NCPUS=$3 (运行时调用的 CPU 核心数)
```

待完成后再 checktst 看是否都已通过。

一般情况下 exam42 和 exam43 会由于 linux 默认的单个共享内存段的最大值太小而不能通过使用

/sbin/sysctl -a | grep shmmax 命令察看默认值，这两个任务需要约 48MB，修改 /etc/sysctl.conf，在里面加入一行 kernel.shmmax = 500000000 将其增加到约 500MB。可以使用命令

/sbin/sysctl -w kernel.shmmax=500000000 来使其立即生效

参考文献：

参考文献 1: <http://hi.baidu.com/sobereva/blog/item/fff7b5fd1e8dd04fd6887db6.html>

参考文献 2: 王涛，量子化学计算程序包GAMESS，科学计算应用软件系列介绍